A machine learning algorithm based on decision trees is called Random Forest Trees (RFT). Machine learning algorithms that do ensemble classification include Random Trees (RT). The term "ensemble" denotes a technique that averages the forecasts of various different base models to provide predictions.
The core idea behind ensemble methods based on randomization is to "incorporate random perturbations into the learning procedure to build multiple alternative models from a single learning set L and then to aggregate the predictions of those models to make the ensemble prediction" (Louppe, 2014). In other words, "growing an ensemble of trees and letting them vote for the most popular class has resulted in significant gains in classification accuracy. These ensembles are frequently grown by creating random vectors that control how each tree in the ensemble grows (Breiman, 2001).

When building a random tree, there are three basic options available. These three considerations are: (1) how to separate leaves; (2) what kind of predictor to utilize in each leaf; and (3) how to introduce unpredictability into trees (Denil et al., 2014). Using a bootstrapped or sub-sampled data set to generate each tree is a typical method for adding unpredictability to a tree. As a result, there are variances among the trees in the forest since each tree in the forest was trained using slightly different data (Denil et al., 2014). The optimal split at a particular node can alternatively be chosen randomly; tests have shown, however, that where noise is relevant, bagging typically produces better results (Louppe, 2014).

"Special attention must be taken so that the resulting model is neither too simple nor too complex," according to the author, when optimizing a Random Trees model. The model is in fact stated to have underfitted the data in the first scenario, i.e., it was not adaptable enough to capture the structure between X and Y. The model is said to be overfit the data in the latter scenario because it is too flexible and captures isolated structures (i.e., noise) that are unique to the learning set (Louppe, 2014).

In order to prevent overfitting, stropping rules must be established to stop a tree from developing before it has too many levels: User-defined hyper-parameters are used to establish stopping conditions (Louppe, 2014). The most popular of these parameters are:
The bare minimum of samples that a terminal node needs to divide
the bare minimum of samples in a leaf node after splitting the terminal node
The maximum depth of a tree, or the number of levels it can reach,
once the Gini Impurity index, which measures the Trees accuracy, falls below a predetermined threshold
To identify the best trade-off, these parameters must be fine-tuned; they must be neither too stringent nor too loose for the tree to be neither too shallow nor too deep (Louppe, 2014).
Breiman (2002) lists the following as some of the essential characteristics of random trees:
It is a very good classifier, with accuracy on par with support vector machines.
As the forest grows, it produces an internal, unbiased estimate of the generalization error.
When up to 80% of the data are missing, it nevertheless retains accuracy thanks to an efficient estimation algorithm.
It has a technique for balancing inaccuracy in data sets with an imbalanced class population.

The generated forests can be saved for use on other data in the future.
It provides an estimate of the variables that are crucial for classification.
Information regarding the relationship between the variables and the categorization is shown in the output that is produced.
It calculates distances between examples that can be used for grouping, finding outliers, or scaling to provide intriguing data visualizations.
Contrary to the Support Vector Machine (SVM), the random trees classifier can typically handle a mix of categorical and numerical variables. As for data scaling, Random Trees are less susceptible to it than SVM, which frequently requires data to be normalized before training or classification. SVM is said to perform better, nonetheless, when the training set is little or uneven. Comparable in computational complexity to SVM, the Random Trees classifier performs better and more quickly with big training sets.